CptS 515 Advanced Algorithms HOMEWORK-5

Sharath Kumar Karnati

NOVEMBER 2023

QUESTION 1: Consider a family H of hash functions: $H = hi : 1 \leq i \leq 8$ Each hi is to map an array of eight bits into its i-th component: hi(a1 $\cdot \cdot \cdot$ a8) = ai. Is H universal? why or why not?

SOLUTION:

Given,
Family H of hash functions , H ={h_i: $1 \leq i \leq 8$}
Here, each h_i is to map an array of eight bits into its ith component.
Lets say that H is universal, then,
The probability of any two distinct input arrays (lets assume a,b) are intersecting if it is less than or equal to 1/M ,for any of the h_i that is choosen randomly from given family H,then ,
if $\forall x \neq y \in U$ and P[h_i[x]=h_i[y]] $\leq$ 1/M
and each h_i belongs to family H as mentioned which implies that here it maps an array of eight bits to its ith component.
Now, as we know that ith component will also be an bit having values of only 0's and 1's , then we can say that M will be equal to 2.
Now lets say for some $x \neq y$, then the probability of that some randomly chosen h_i which belongs to H ,which maps x and y to the same value will be equal to the probability of the i th bit which will be same for the both x and y.
Now lets take the x and y values as ,
x = 00000000 and y = 00000001, here both of them are differ by 1 digit.
Now, we will have seven hash functions that are mapping them into same slot.
i.e., P[h_i[x]=h_i[y]] $\leq$ 7/8 ,
here 7/8 > 0.5 , as we have take M=2 in this case.
So, it does not follow the definition as it states that it should be $\leq$ 1/2.
Hence , this is contradicting our definition.
Therefore, H is not a universal hash function.

QUESTION 2:

Here is a classic example of universal family of hash functions. Let M be a prime number and, as usual, [M] = 0, 1, $\cdot \cdot \cdot$, M 1. Consider the following family of hash functions: hr(x) = (r $\cdot$ x mod M), where r, x [M]k (where k is a given constant like 10), and r $\cdot$ x = i rixi. Show that the family of hash functions (for the given k) is universal.

SOLUTION:

Given, M is a prime number and [M] = {0,1,....,M-1}
hr(x) = (r. x mod M), where r,x∈[M] k where k is a given constant like 10.
and r . x = Σ i rixi,
We know that, for a hash function to be universal, if and only if the probability of any two distinct input arrays (x,y) are intersecting then, if it less than or equal to 1/M, for any h_r choosen from the family H, which is
if $\forall x \neq y \in U$ and P[h_r[x]=h_r[y]] $\leq$ 1/M .
now to prove the given hash function as universal, we are gonna consider the condition that we will have x and y which can absorb k values and we are gonna state that they will have a different first component which is at position 0 , i.e.,$x_0 \neq y_0$
Now, we are gonna consider the condition of intersection :
i.e., h_r[x]=h_r[y]
$\Sigma_{i=0}^{k}$ $r_i.x_i$ (mod M)=$\Sigma_{i=0}^{k}$ $r_i.y_i$ (mod M)
$\Sigma_{i=0}^{k}$ ($r_i.x_i$ - $r_i.y_i$) $\equiv$ 0 (mod M)
$r_0(x_0\text{-}y_0)$ + $\Sigma_{i=0}^{k}$ ($r_i.x_i$ - $r_i.y_i$) $\equiv$ 0 (mod M)
$r_0(x_0\text{-}y_0)$ $\equiv$ - $\Sigma_{i=0}^{k}$ ($r_i.x_i$ - $r_i.y_i$)
Now from the number theory , we can state that for each M which belongs to {0,1,....,M-1} there always be an multiplicative inverse M^{-1} so that $M.M^{-1} \equiv$ 1(mod M).
Now, we are gonna divide (x_0-y_0) on both sides, then we will get:
$r_0 \equiv$ - $\Sigma_{i=0}^{k}$ (r_i. (x_i - y_i). (x_0-y_0)$^{-1}$
Now, from the above equation , we can conclude that this can be only be valid when we have exactly one value of r_0 and this is only possible when we have a probability of 1/M . As we have probability collision of atmost 1/M then it will be an universal.
Therefore, we can state that the given is a universal hash function.

QUESTION 3: So far, what we have learned about hasing is to hash an array of numbers into one number (e.g., locality sensitive hashing). Can you suggest a way to hash a graph into a number (which could be a real number)?

SOLUTION:

Now, for conversion of hash a graph into number , we are going to use the concept of reduction, in which we are gonna use any of the hash function which hashes a array of numbers into a one number with the help of matrix representation of a graph which converts it into a vector.
Methodology :

- Firstly, we are gonna consider the given graph (G) . Then we will calculate the Laplacian matrix(L) for the graph because we know that it contains the data of adjacency(A) and degree(D) of a graph with it . (L = D - A)
- Now, we will do the vectorization of the matrix L by adding each row step-wise of the matrix(L) to a new list (K) .
- Then, now we will use some of the locally sensitive hashing techniques that will help to hash the list (K) into a number.

Therefore , by using this methodology we can hash an array of numbers into one number.

QUESTION 4: Randomized quicksort is a Las Vegas algorithm where the first step is to create a random permutation of the input arrayof numbers before the second step of running quicksort. Now, we assume that we have a high quality psuedo random generator r(n) that will generate a random number in 1..n. Please show how to generate a "random" graph with 5 nodes.

SOLUTION:

Now, for this we are gonna use the ERDOS-RENYI model method to generate a random graph with 5 nodes. In this, there are two methods to use this model and I am gonna explain the both:

Methodology 1 :

- Firstly, we will use the ERDOS-RENYI model G(n,m), here n represents the number of the nodes and m represents the total number of the edges of the G.
- Now, we are gonna assume that there will be only one edge between the two nodes.
- Now, from the given case, we have n=5 and $m_max = n * (n-1)/2$ which will be equal to 10 (m=10).
- Now, we will use the random generator r_n to pick a number(n) randomly from (1,2,.....,10)
- Now, we will choose a node pair randomly without an edge, then we will add an edge them, then n will be n=n-1 .
- Then , we are gonna repeat the above step until we have n=0 , then we will finally get the random graph with 5 nodes.

Methodology 2 :

- Firstly, we will use the ERDOS-REYNI model G(n,p), here n is number of nodes and p represents the probability of each edge that are occured independently.
- Here are also we are gonna assume the same as we will have only one edge between 2 nodes.
- We will use the random generator (r_n) to pick a number(n) from (1,2.....100) , so that p=n/100.
- Now, we are gonna label all pairs of the nodes. Then, we are gonna choose a pair that we haven't used before.
- Then, now again use the (r_n) to pick a random number n from (1,100), so that r=n/100.
- If r > p, then we are gonna add an edge between those two selected nodes, if not nothing needs to be done.

- Then, we are gonna repeat the above step until all the nodes pairs are choosen, then finally , we will get the desired random graph with 5 nodes.

QUESTION 5: Mr. X drives on I-90 all the way from Pullman to New York (Let's assume that Pullman is Spoakne). On his car, there is a device that can suggest all the interesting places nearby that Mr. X might visit (and spend some money at these places of course). These places are stored in a set S and will be updated automatically while Mr. X is driving. Please suggest a way to implement the S so that Mr. X can query (e.g., "Is there a restrant nearby?", etc.). You shall use Bloom filter to store S. Feel free to look up papers on the Internet. 1

SOLUTION:

For this problem, we are mainly gonna use the two key operations which are deleting an element from the given set and retrieving it by the keys. So now, we are gonna use the invertible bloom filter as the static bloom filter does not allow these two operations.
Operations supported by invertible bloom filters:

- INSERT(a,b) : Here it inserts the key-value pair(a,b) into X. This will always work assuming that all keys are different. (a,b) $\in$ X, which we will assume that for rest of the
- DELETE(a,b) : This will delete the key-value pair (a,b) from the X, this will always work assuming that given (a,b) $\in$ X, which we will assume that for rest of the
- GET(a) : It will return the value of b which is a value of key a in the X.Then, if b=null is returened, then the (a,b) $/\in$ X for any value of b. This condition may fail with a low probability(but constantly) by returning a "NOT FOUND" error. This is because there exists or does not exists the key-value pair(a,b) in X that we are looking for.
- LISTENTRIES() : This is generally used to make list of all the key-value pairs that are in X. This may sometimes return a partial list with an error like "LIST INCOMPLETE". This is used to list all entries of the given set.

Now, here Lets assume that the invertible bloom look up table (IBLT) data structure X, it is a randomized Data structure which will store the set of the given key-value pairs.
Now for this problem, firstly to store the all interesting locations of Mr.X in the invertible bloom filter as a key value pairs , we are gonna use the "INSERT" condition.
Then , we are gonna use the "DELETE" AND the "INSERT" operations to constantly update the set S as the location of Mr.X keeps on changing as he is driving.
Then, to handle the other queries like (eg: Is there a restaurant nearby ?) , we will use the "GET" operation for this.

Now, all these operations are done by the Invertible bloom filters, which will use a three component list. This includes the "count field" which counts the number of entries that are mapped to a particular index, then, "a key field" which does the sum of all the keys that are mapped to a particular index and finally " a value field" which does the sum of all the values that are mapped to a particular index.

REFERENCES:

refrences used click here

QUESTION 6: We know many ways to hash an array of integrs into a number. However, hash itself is loosy — that is, the function many not be one-to-one. Can you suggest a way to hash an array of 10 bits into a number such that a. the hash is one-to-one, and, b. the hash is locality sensitive (i.e., when the Hamming distance between two such arrays of 10 bits is small, then so is the distance between their hash values). (I have a terrific way to do this — but I wont tell you. You shall figure out your own ways to do this. This problem concerns a lot of fundamental applicational problems in computer science.) 2

SOLUTION:

Given, two arrays = a and b, of 10 bits.

METHODOLOGY:

- Firstly, we will calculate the hamming distance between the 2 given arrays D(a,b).
- Then, we will now perform the XOR operation on (a $\oplus$ b), and then we will do the counting of the total number of 1's from the string that we will get. Then, D(a,b) $\in$ (1,2.....,10).
- Now, we will consider the hash function for the hamming distance that will map a d-dimension bit vector to its value on a random coordinate. Then, we will get:
  Prob(h(a) = h(b)) = 1 - D(a,b)/d .
- We know that , when the distance between two strings is smaller then it will be more likely for the hash functions to be equal.
- Now , we will define the hash family(H) under conditions like : for a parameter(c) > 1 and probability value p_1 will be greater than p_2, and the hamming distance D(a,b) $\in$ (1,2,....,10), and then a Hash family will be (r, c_r,p_1,p_2) a local sensitive , then $\forall$ q,a,b
  Then, if D(a,b) < r ,then Prob(h(a) = h(b)) $\geq$p$_1$ and then if D(a,b) > cr, then Prob(h(a) = h(b)) $\leq$p$_2$ and the probability where h $\in$ H is choosen randomly.
- Now, we will build the hash family accordingly and super bit locality-sensitive hashing.
- We will select the h $\in$ H randomly untill we get,
  $\Sigma_{j=0}^{N-1}\ n_j^2 \leq 4N$
  Now, for each jth slot with the ($n_j > 1$), Now, we will select the h_j so that the n_j keys will be hashed into the n_j^2 slots without the collision.

Now by using this methodology we can hash a array into number with the given conditions.
